Random

Objective

In this projectlet, we venture into random numbers. In particular we will explore the ‘quality’ of random number generators available. In most explorations of this kind, a great way to begin is to visualize using graphs/plots. This will be our approach.

Most random number generation support from libraries will yield a deterministic sequence based on a seed provided. This is quite appropriate during development. However in production this may not be satisfactory. A source with “true” randomness might be more appropriate. For example, many encryption algorithms will depend on a random sequence of numbers as the primer. In this exercise, we will use a cryptographically secure random number source in addition to the basic random number sources.

Specifications

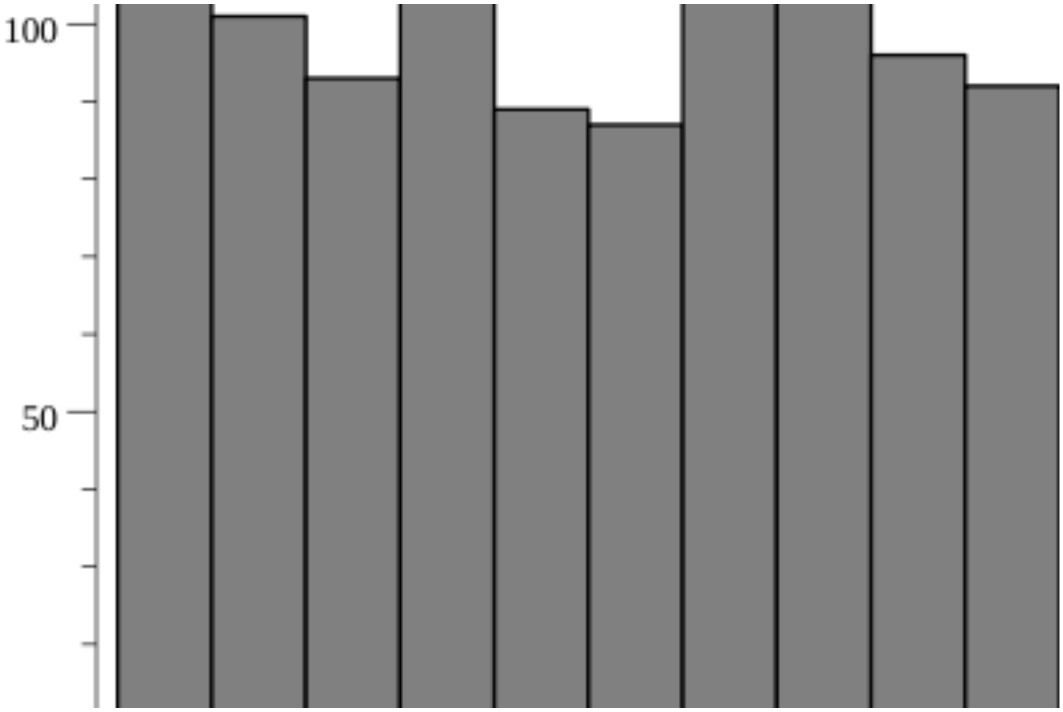
Command	Switch	Description
uniform		Uniform random numbers
		Arguments are: Minimum maximum Default: 0.0 1.0
normal		Normally distributed random numbers
		Arguments are: Mean Standard Deviation Default: 0.0 1.0
	—crypto	Use a cryptographically secure random number source
	—output	Output file name. Creates a png file.
	—series	Output plot is a series. By default it will be a histogram (configure with —slices)
	—samples	Count of number of samples. Applicable to all distributions. Default 1000
	—slices	Number of bins for histograms
	—seed	Seed the series with this number. Default 1729. This is not used if the —crypto switch is used
	—table	Generates a table of the numbers generated - in addition to the plot. Value is the filename.

Example usage

```
bin/random --help

     This utility generates random numbers and plots the data as a series or
as a histogram.

Usage:
  random [command]

Available Commands:
  help        Help about any command
  normal      Normal distribution
  uniform     Uniformly distributed random variables
  version     Report the version of the application

Flags:
  -C, --crypto          use cryptographic random source
  -h, --help            help for random
  -o, --output string   output file name (default "plot.png")
  -s, --samples int     number of samples (default 1000)
  -d, --seed int        seed for random numbers (default 1729)
      --series          generate time series plots. default is histogram
  -c, --slices int      number of slices - for histograms (default 10)
  -t, --table string    tabular output filename
      --verbose         be verbose
  -v, --version         version for random

Use "random [command] --help" for more information about a command.
```

Uniform Distribution Examples

An observation may be - repeated generation of this histogram will result in identical results. The reason for this is that the linear congruential generators typically used for this function produce identical results for a given seed.



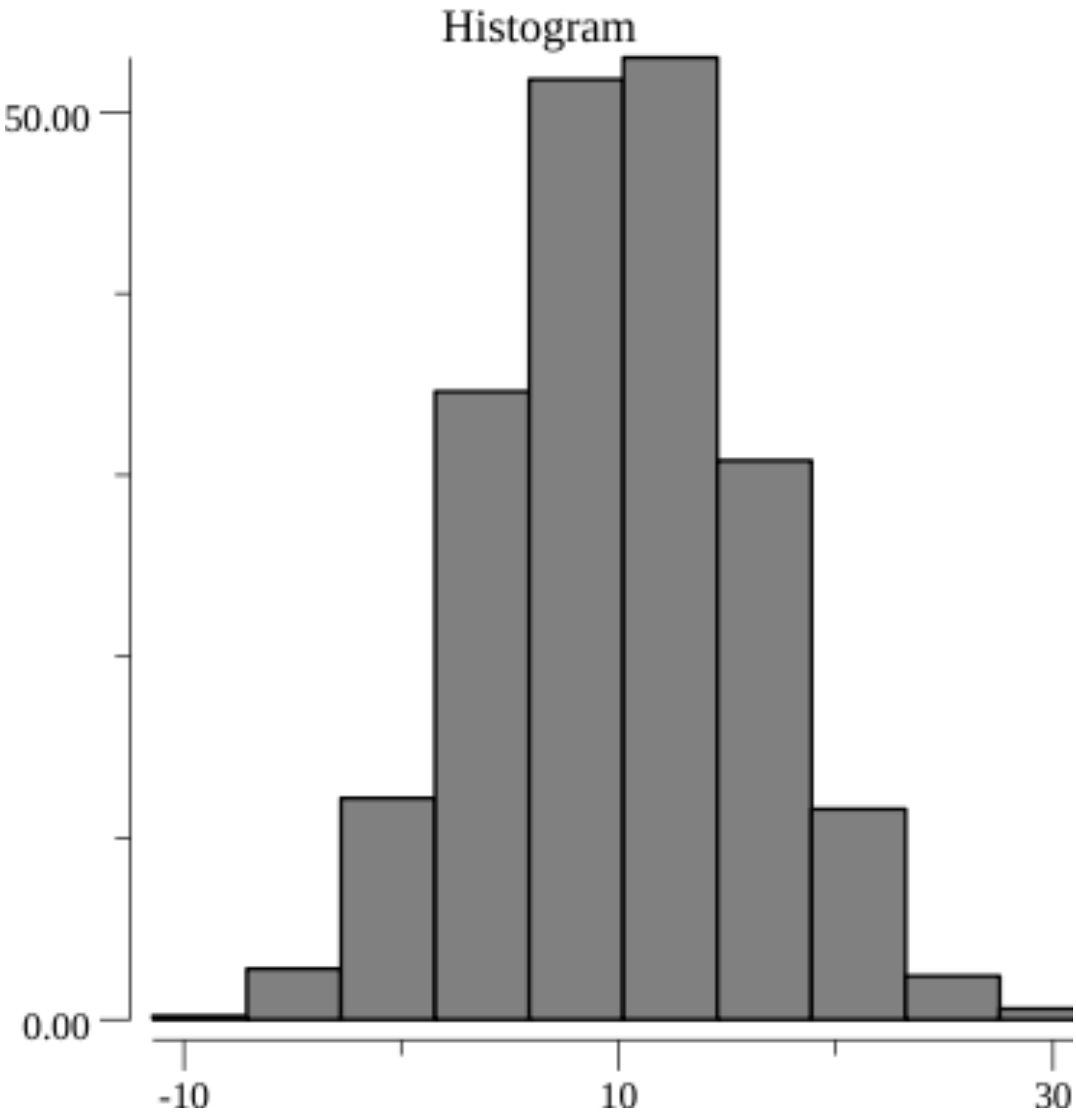
UNIFORMLY DISTRIBUTED 1000 RANDOM NUMBERS - HISTOGRAM

Normal Distribution Examples

With the command:

```
bin/app normal --output histnorm.png 10.0 6.0
```

Ie mean of 10.0 and standard deviation 6.0 generates the following histogram

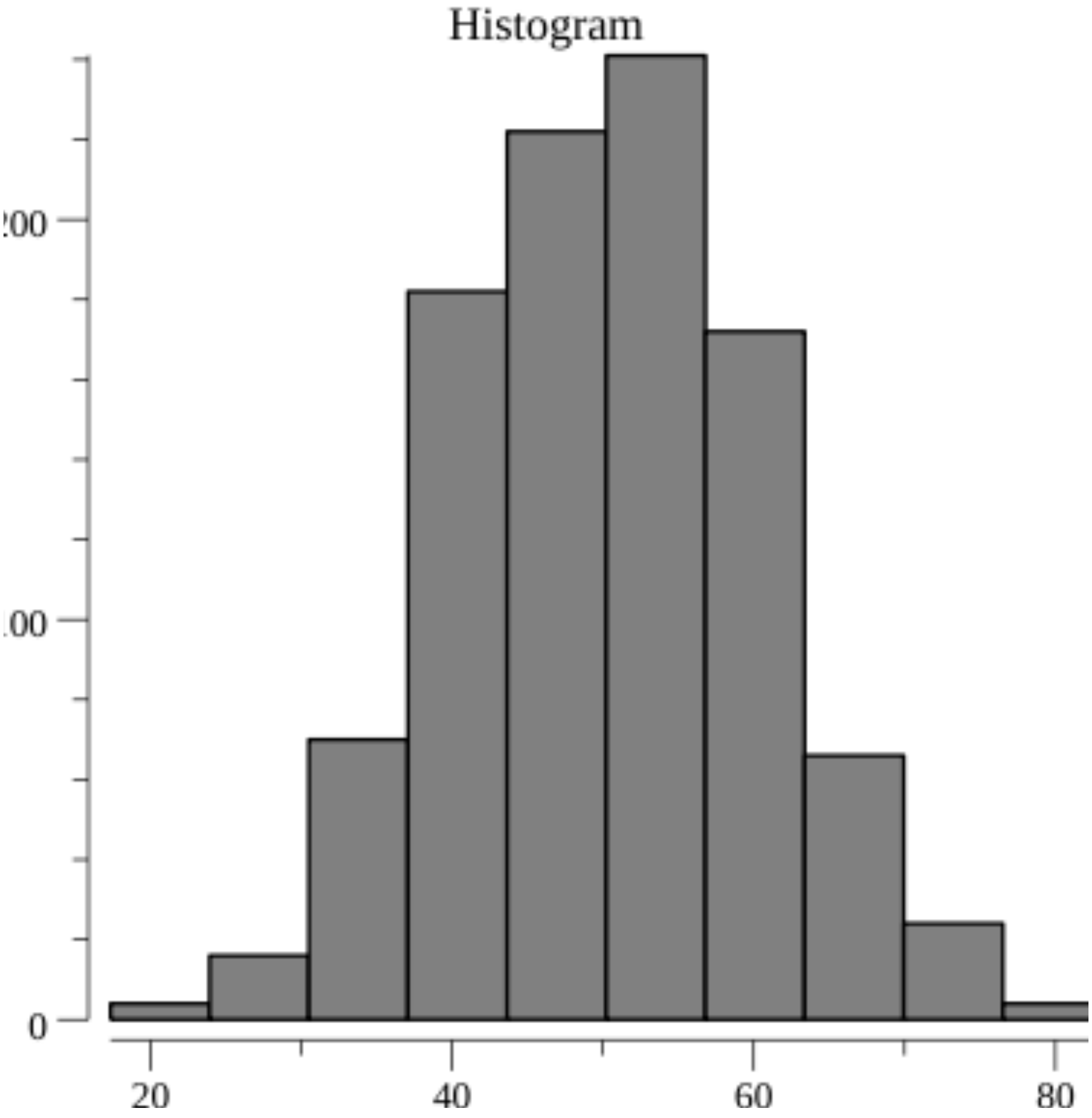


NORMAL DISTRIBUTION - MEAN 10.0 STDEV 6.0

Crypto Normal

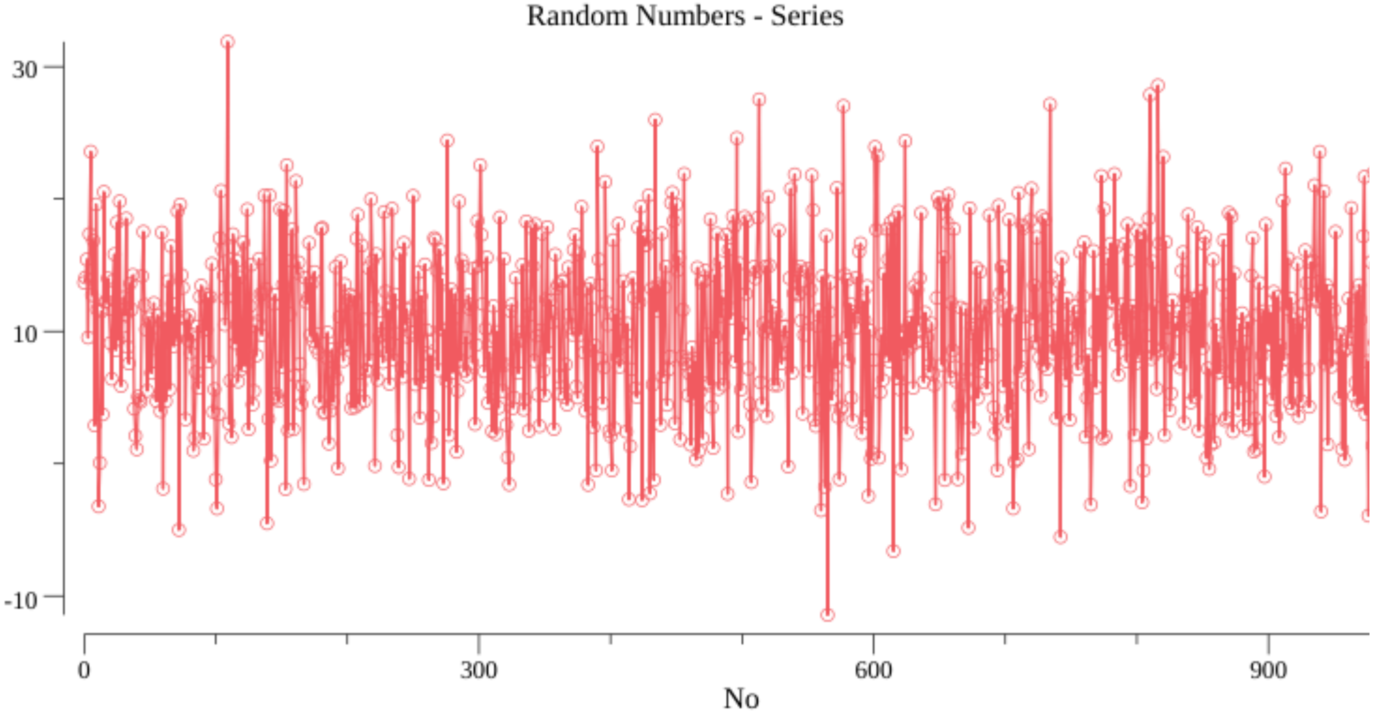
In this example we ask for a crypto source. Running this command repeatedly generates different plots/histograms.

```
bin/random normal -t table --samples 1001 --crypto 50.0 10.0
Dumping values to table
```



NORMAL DISTRIBUTION - CRYPTO SOURCE

Series plot of Uniform random numbers



RANDOM NUMBERS AS A SERIES PLOT

Implementation Examples

Go language example

https://gitlab.com/RajaSrinivasan/random.git

Potential Improvements

More distributions	Other continuous distributions such as Weibull.
Discrete Random numbers	Other forms of graphing may be more appropriate
Evaluate quality of the series	Analyze the series to find out closeness to the requirement. Is the quality better with larger number of samples?